EVTGEN_trigger_ADC for KIT_T2G-B-H_LITE

Customer training workshop

Q3 2024

Scope of work

- This code example demonstrates how to use the TRAVEO™ T2G MCU event generator (EVTGEN) resource to trigger ADC conversion in Active power mode. In this example, the event generator is configured to trigger an ADC conversion every second, and when the ADC conversion is complete, print out the ADC result via UART.

- **Device**
  - The TRAVEO™ T2G CYT4BF8CDS device is used in this code example

- **Board**
  - The TRAVEO™ T2G KIT_T2G-B-H_LITE board is used for testing

Introduction

- **Event Generator (EVTGEN) has the following features:**
  - CPU-free triggers for device functions
  - Reduces CPU involvement in triggering device functions; reducing overall power consumption and CPU bandwidth
  - 16 comparators for each Deep Sleep and Active mode to generate interrupts and triggers
  - 32-bit counter, one each for Deep Sleep and Active mode for comparators
  - Individual configurable thresholds for each comparator
  - Deep Sleep and Active mode clock sources for counters
  - Jitter-free initiation of specific device functionality
  - One Deep Sleep and one Active mode interrupt for CPU
  - Supported in Active, Sleep, LPActive, LPSleep, and Deep Sleep power modes

- **ADC has the following features:**
  - SAR ADC core
    - 12-bit resolution with a maximum sample rate of 1 Msps
  - 32 logical channels with the same capabilities
  - Each logical channel can select input from
    - 32 analog input pins
    - Diagnostic signals
    - Analog input pins of other ADC units
    - Support for external mux (three select bits)
    - AMUXBUSA/B

Introduction (contd.)

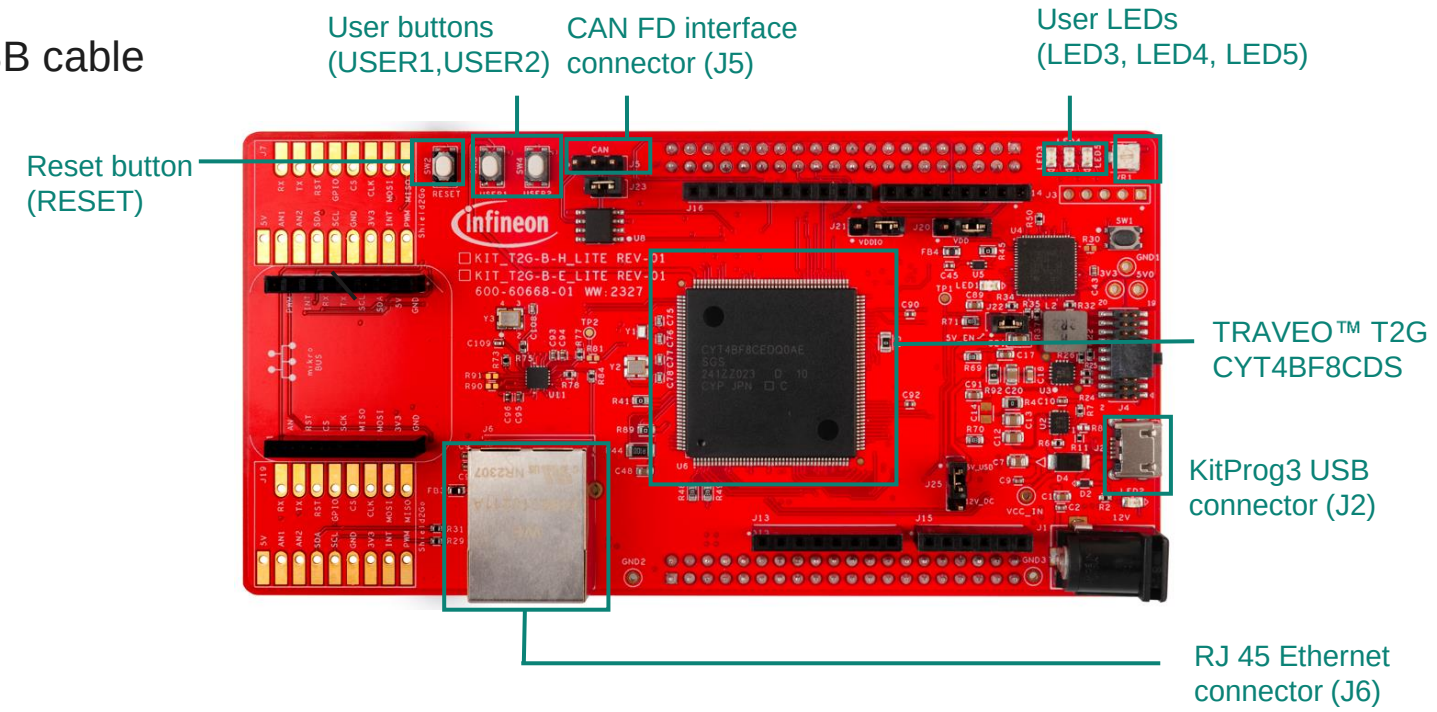
- **ADC has the following features:**
  - Scans triggered by timer, software, continuous, pins, or system triggers
    - Multiple ADC units can be triggered by the same trigger to ensure lock-step operation
    - Triggers can be cleared by software
    - Optional debug pause
  - Double buffering of output data
  - Programmable sample time for each channel
  - Programmable post processing options for each channel
    - Sign/zero extension to 16-bit
    - Left/right alignment
    - Averaging: first order accumulate and dump, up to 256 samples
    - Programmable right shift
    - Range detection: below/above threshold, in/out-side range
    - Pulse detection: programmable positive and negative event counters
  - Channels can be individual or grouped
    - Flexible grouping: from 32 groups with one channel to one group with 32 channels
  - Group scans are dynamically scheduled by the hardware
    - Eight priorities, programmable per group
    - Four preemption types: resume, restart, cancel, or finish
    - Optional automatic idle power down

Introduction (contd.)

- **ADC has the following features:**
  - Interrupt generation
    - Group scan done
    - Group scan done overflow detect
    - Group scan canceled
    - Per channel range detect
    - Per channel pulse detect
    - Per channel pulse/range overflow detect
  - Output trigger generation per channel
    - Data ready/completion (each channel can trigger DW transfer)
    - Range violation detected
  - Digital and analog calibration available
  - Programmable offset and gain calibration
    - Non-intrusive background recalibration
    - Coherent calibration update
  - Support for diagnostic measurements including broken wire detection. This includes:
    - ADC sampling capacitor preconditioning feature
    - Selectable current source or sink on selected ADC input while sampling
    - Support for LED diagnostics
  - On-chip temperature sensor and power monitoring

Introduction (contd.)

- This code example has been developed for the KIT_T2G-B-H_LITE board
- Connect the PC to the board using the provided USB cable through the KitProg3 USB connector (J2)



Implementation

- This code example demonstrates how to use the event generator in Active power mode. The counter clock of the event generator block is configured to 1 MHz, and the active compare value is configured to 1000000. When the active counter is greater than or equal to the active compare value, it will generate the active trigger event to trigger SAR ADC conversion and interrupt. In the event generator interrupt handler, it will update the active comparator value and generate events every second to trigger ADC conversions. When SAR ADC conversion is complete, print out the ADC result via UART.

- **Follow these steps to configure this code example:**
  - STDOUT setting
  - ADC initialization
  - ADC interrupt configuration
  - Event Generator interrupt configuration
  - Event Generator initialization
  - Start Event Generator
  - Event Generator comparator structure initialization
  - Update active comparator value
  - Get ADC conversion result

Implementation (contd.)

- **STDOUT setting**
  - The ***cy_retarget_io_init ()*** function initializes the GPIO for UART
    - Initializes P0.1 as UART TX, P0.0 as UART RX (these pins are connected to the KitProg3 COM port)
    - The serial port parameters change to 8N1 and 115200 baud

- **ADC initialization**
  - Call the ***CY_SAR2_Init()*** function to initialize the ADC channel
    - Initialize the ADC channel 0 to use pin P6.0 as input

- **ADC interrupt configuration**
  - Call the ***Cy_SAR2_Channel_SetInterruptMask()*** function to configure the channel interrupt
    - Specify ***CY_SAR2_INT_GRP_DONE*** as an argument to generate an interrupt when the conversion is completed
  - Configure interrupt in the ***CY_SysInt_Init()*** function
    - Set the interrupt source (ADC channel 0), interrupt priority (7), interrupt vector, and the ISR (***adc_int_handler()***)
  - Then, clear the IRQ request of the configured interrupt by ***NVIC_ClearPendingIRQ()***[1], before enabling ***IRQ by NVIC_EnableIRQ()***[1]

- **Event Generator interrupt configuration**
  - Configure interrupt in the ***CY_SysInt_Init()*** function
    - Set the interrupt source (EVGEN0), interrupt priority (7), interrupt vector, and the ISR (***evtgen_isr()***)
  - Call the ***Cy_EvtGen_ClearInterrupt()*** function
    - Set the EVTGEN0 bit to '0' to clear the interrupt
  - Then, clear the IRQ request of the configured interrupt by ***NVIC_ClearPendingIRQ()***[1], before enabling IRQ by ***NVIC_EnableIRQ()***[1]

[1]: The CPU interrupt enable and NVIC operation instructions are provided by Cortex microcontroller software interface standard (CMSIS) with intrinsic functions.

Implementation (contd.)

- **Event Generator initialization**
  - Call the ***Cy_EvtGen_Init()*** function to initialize the event generator
    - Initializes Event Generator parameters (clock source frequency in Active/Deep Sleep power mode, EVTGEN customized period, ratio control mode and specific dynamic mode)
    - See also ***cy_stc_evtgen_config_t*** for parameter details

- **Start Event Generator**
  - Call the ***Cy_EvtGen_Enable()*** function to start the event generator
    - Enable the Event Generator
  - Call the ***Cy_SysLib_DelayUs()*** function to set the delay bit and wait for the counter to complete initialization
    - Waiting for 625 µsec
  - Call the ***Cy_EvtGen_GetRatioStatus()*** function check to determine if the ratio status is valid during hardware control ratio
    - Get VALID bit of EVTGEN0_RATIO_CTL register
  - Call the ***Cy_EvtGen_GetCounterStatus()*** function to check if the event generator counter status is valid
    - Get VALID bit of EVTGEN0_COUNTER_STATUS register

- **Event Generator comparator structure initialization**
  - Call the ***Cy_EvtGen_InitStruct()*** function to initialize the event generator comparator structure
    - Initializes the Event Generator parameters (functionality comparator structure, condition for start trigger/interrupt, making period of interrupts/triggers and the making period of interrupts during Deep Sleep)
    - Refer to ***cy_stc_evtgen_struct_config_t*** for parameter details

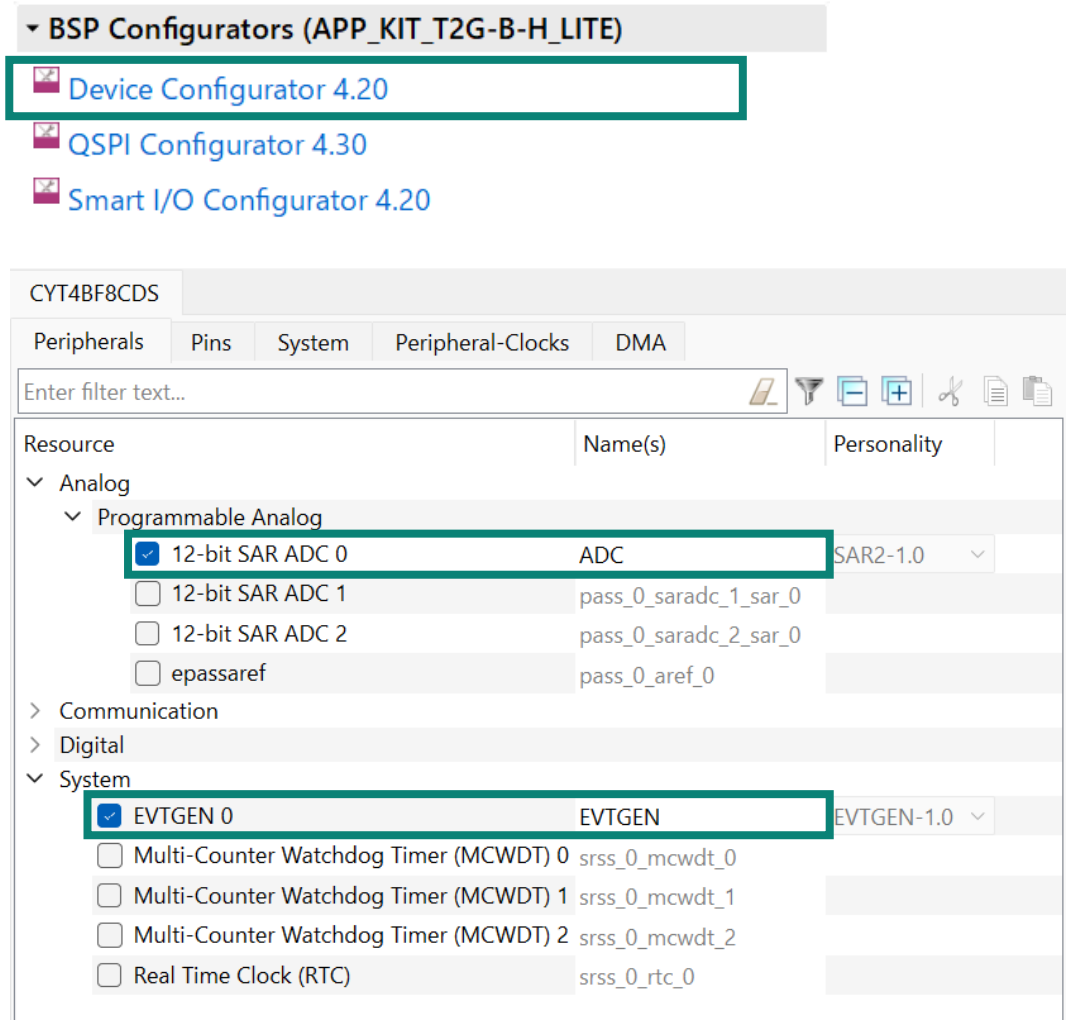
Implementation (contd.)

- **Update active comparator value**
  - Check that the interrupt source is EVTGEN0 with the return value of the ***Cy_EvtGen_GetStructInterrupt()*** function
    - Get interrupt flag of EVTGEN0
  - Call the ***Cy_EvtGen_ClearStructInterrupt()*** function to clear the interrupt factor
    - Clear interrupt flag of EVTGEN0
  - Call the ***Cy_EvtGen_UpdateActiveCompValue()*** function to update the active comparator value
    - Update active comparator to initial value (1000000 = 1 s); this can be modified to change the ADC conversion cycle

- **Get ADC conversion result**
  - Call the ***Cy_SAR2_Channel_GetResult()*** function to get the ADC conversion result and status
  - Call the ***Cy_SAR2_Channel_ClearInterrupt()*** function to clear the interrupt factor
    - Clear interrupt flag of specified ADC channel (channel 0 of ADC0)

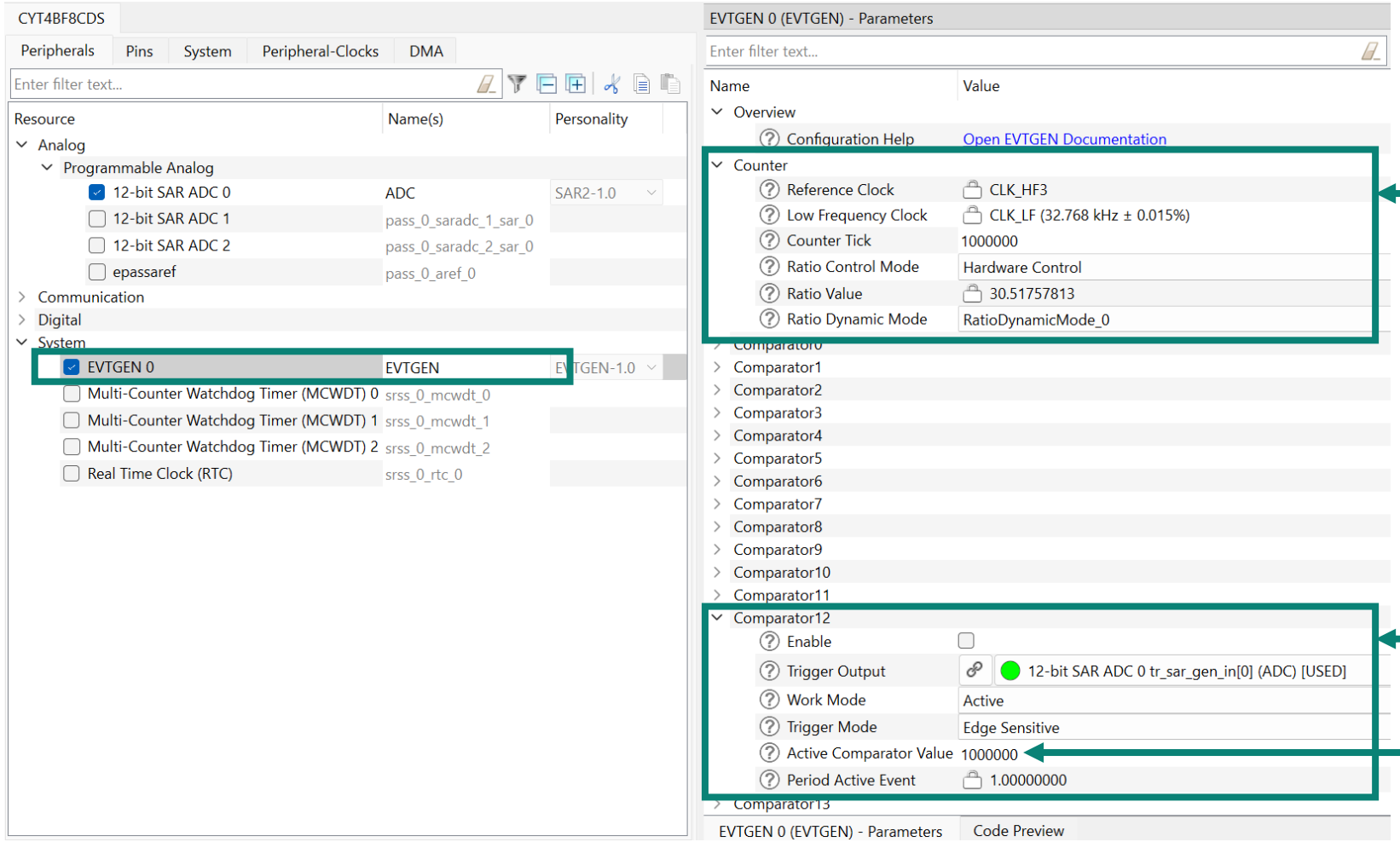
Implementation (contd.)

- **Configure Event Generator and ADC**
  - You can change Event Generator and ADC configuration by Device Configurator
    1. Open Device Configurator
    2. Select **Peripherals** tab
    3. Select **12-bit SAR ADC 0** from Programmable Analog under Analog
    4. Select **EVTGEN 0** from System under Resource



Implementation (contd.)

– **Event generator configuration**

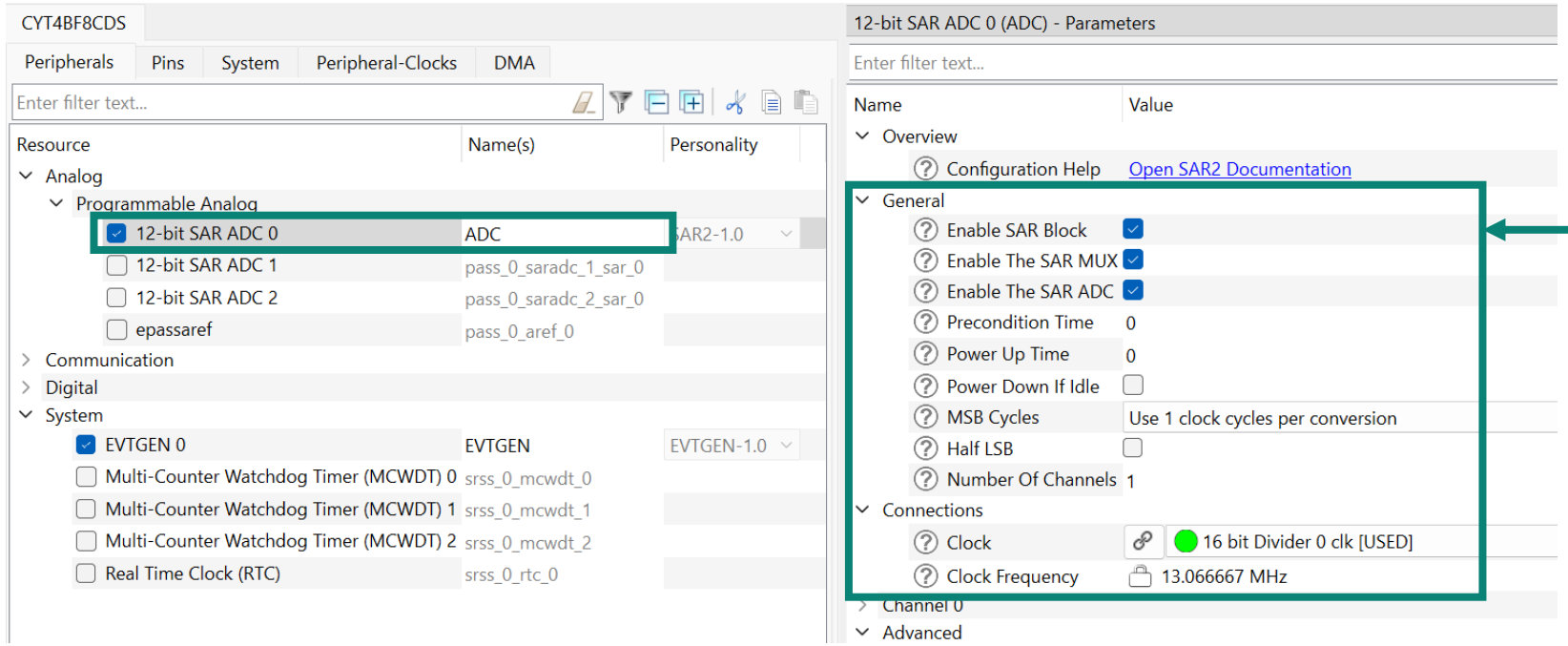


Event Generator setting in this example

You can change the conversion cycle (default one second)

Implementation (contd.)

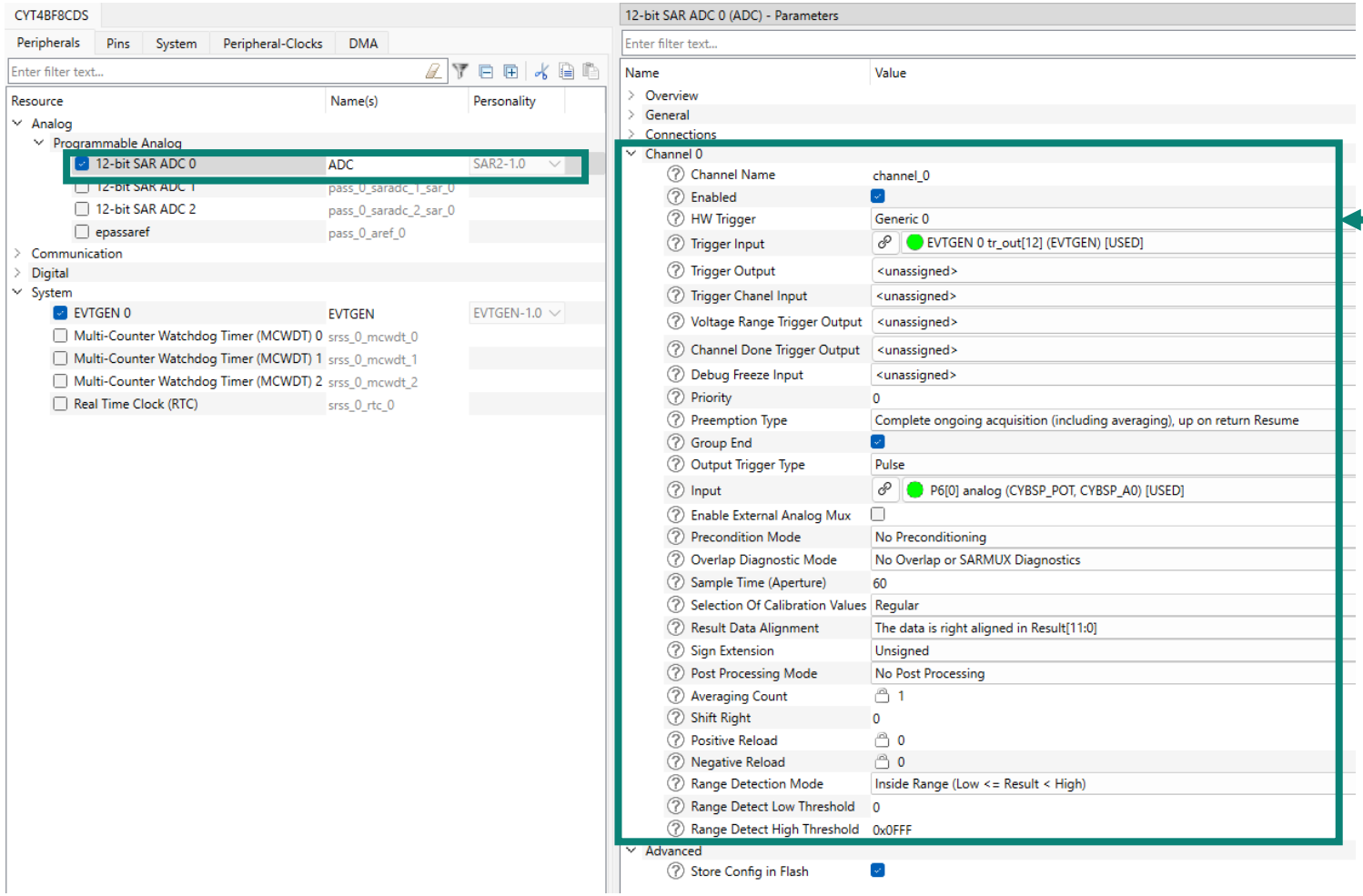
– **Event Generator configuration**



ADC general and clock setting in this example

Implementation (contd.)

– **Event Generator configuration**

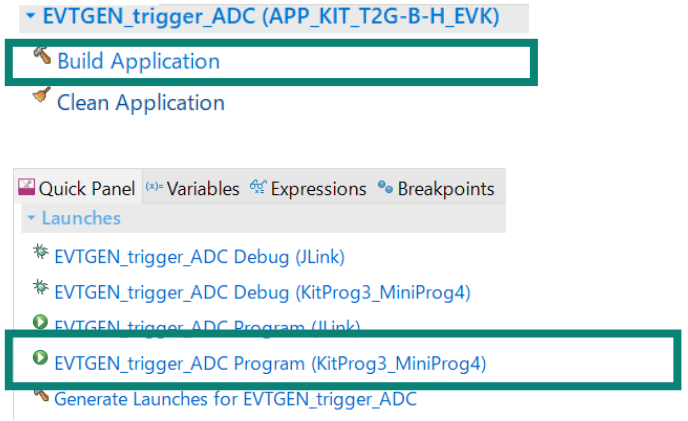


ADC trigger setting in this example:
- Trigger input: EVTGEN 0
- Analog Input: P6_0

Compiling and programming

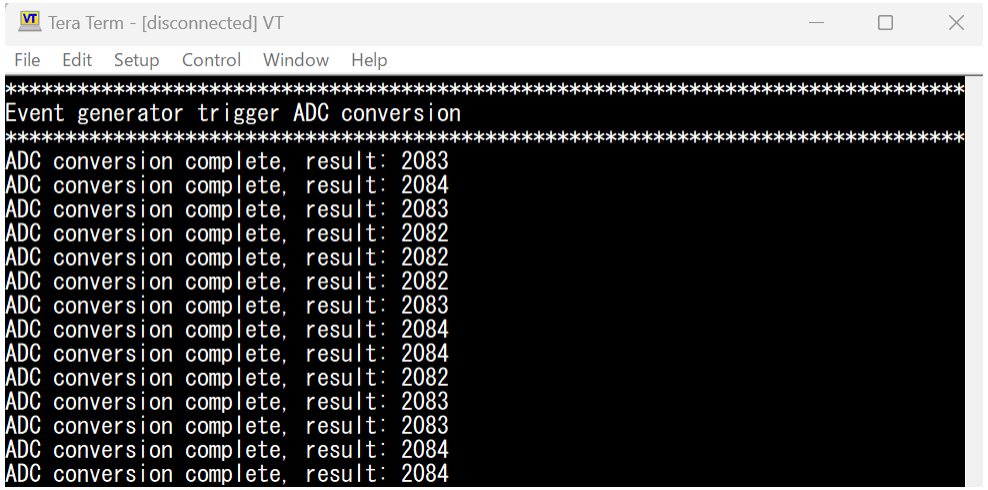
1. Connect to power and USB cable
2. Use Eclipse IDE for ModusToolbox™ software for compiling and programming
3. For compilation:
   a. Select the target application project in the Project Explorer
   b. In the Quick Panel, scroll down, and click **Build Application** in EVTGEN_trigger_ADC (KIT_T2G-B-H_LITE)
4. Open a terminal program (such as Tera Term) and select the KitProg3 COM port. Set the serial port parameters to **8N1** and **115200 baud**.
5. For programming:
   a. Select the target application project in the Project Explorer
   b. In the Quick Panel, scroll down, and click **EVTGEN_trigger_ADC Program (KitProg3_MiniProg4)** in the Launches

KitProg3 USB connector (J2)



Run and test

1. After programming, the application starts automatically.
2. Ensure that input voltages are provided at the analog input pin P6.0. This pin is connected to the potentiometer (VR1).
3. Rotate the potentiometer to change the ADC input voltage, and the results are printed out every second in the terminal window.
4. A message is displayed on the UART terminal as shown in the figure.



References

- **Datasheet**
  - CYT4BF TRAVEO™ T2G 32-bit Automotive MCU based on Arm® Cortex®- M7 dual
- **Architecture reference manual**
  - TRAVEO™ T2G Automotive MCU body controller high architecture reference manual
- **Registers reference manual**
  - TRAVEO™ T2G Automotive MCU: TVII-B-H-8M body controller high registers reference manual
- **PDL/HAL**
  - Peripheral driver library (PDL)
  - Hardware abstraction layer (HAL)
- **Training**
  - TRAVEO™ T2G training

Revision History

Revision	ECN	Submission Date	Description of Change
**	7840285	2022/11/24	Initial release
*A	8085468	2024/10/29	Replaced development board from KIT_T2G-B-H_EVK to KIT_T2G-B-H_LITE


Infineon